



# UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
United States Patent and Trademark Office  
Address: COMMISSIONER FOR PATENTS  
P.O. Box 1450  
Alexandria, Virginia 22313-1450  
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
-----------------	-------------	----------------------	---------------------	------------------

09/998,848

11/15/2001

Kenneth Y. Ogami

CD01177M

6884

60909

7590

12/10/2009

CYPRESS SEMICONDUCTOR CORPORATION  
198 CHAMPION COURT  
SAN JOSE, CA 95134-1709

EXAMINER

VO, TED T

ART UNIT

PAPER NUMBER

2191

MAIL DATE

DELIVERY MODE

12/10/2009

PAPER

**Please find below and/or attached an Office communication concerning this application or proceeding.**

The time period for reply, if any, is set in the attached communication.



### **DETAILED ACTION**

1. This action is in response to the amendment filed on 08/12/2009.

Claims 1-14, 16, 26-30, 36-38 are pending in this application.

### ***Response to Arguments***

2. Applicants' arguments in the Remarks section filed on 08/12/2009 have been respectfully considered. Applicants argued Bindra does not teach or suggest the use of template files; Applicants argued Hamblen teaches a C compiler that automatically generates a code generator, but does not teach using a template file for generating code.

Examiner respectfully disagrees. First of all, the recitation "a template file" in the claim phrase, "*wherein constructing the computer-generated source code comprises substituting said user-specifiable information for generic information in a template file*", show the recitation is merely included; it appear does not do anything in the claim. Secondly, using a template file for generating code can not present a patentable feature. By standard, every data file readable via a computer screen is in a certain type of file extension. For example, assembly file generated by a compiler has the extension .asm; generic text has the extension .txt; standard web page has extension .htm, etc. Any Windows embedding word document will provide a blank template for writing/editing texts. Therefore using a template for specifying generic information is merely conforming to a common standard. Without having a file template worked as a basis data holder, data will go to nowhere or can not be existed. Thus, there is no computer.

Art Unit: 2191

In Bindra reference, it does not have the term “template file”; however, it must be inherently used behind the tool shown in Figure 4. For example, see p. 1:

By allowing the user to work with menus and graphical icons on-screen, this Windows-based IDE permits a user to drag and drop higher-level functions in appropriate boxes and create the end system in minutes. Once the system configuration is finalized, it's stored in the flash memory on-chip. Upon power-up, the contents of the flash are transferred into register space that holds the configuration information. Also, this system configuration can be easily and quickly modified as the system requires change.

The paragraph does not mention any word “template”; however, configuration data for a virtual block “a flash memory on-chip” is transferred into a register space. This space holds the configuration information of “the flash”. When the IDE generates execution, the configuration data in the space will be used by the tool compiler to convert into assembly language.

On the other hand, with the suggestion from viewing the Hamblen it shows a programmable on chip design process (p. 36, Figure 11) using a CAD tool that takes design virtual blocks and constructs source code. The source code in form assemble or machine language that is automatically generated by a C compiler to mapped on to the Virtual blocks designed from the CAD tool (See Figure 1: VHDL Design entity (i.e. Virtual blocks) connected with automatically generated code generated by a C Compiler, mapped to virtual blocks at gate levels in the CAD tool). The template is shown under the extension of a file. Texts in Figure 7, 9, 10, are held in template files, etc.

Therefore, the adding limitation in claims, especially for “a template” does not make the claim patentable difference from the prior art. The broad term “a template” is inherently suggested in Bindra as of one the space, where a space holding configuration data of a virtual block as shown in the p. 1.

Art Unit: 2191

Claims 36 and 37 are rejected as being obviousness over the prior arts of record in view examiner' official notice. Examiner suggests the amendment in combining the claims 36 and 37 into claim 1 will result an allowance subject matter.

***Claim Rejections - 35 USC § 103***

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

A person shall be entitled to a patent unless –

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

4. Claims 1-14, 16, 36-37 are rejected under 35 U.S.C. 103(a) as being unpatentable over Bindra, "Programmable SoC Delivers A New Level Of System Flexibility", 2000, in view of Hamblen, "Rapid Prototyping using Field Programmable logic Devices", 6-2000.



Art Unit: 2191

another virtual block; and thus, if depict a selected block such as a module of Figure 1 or another simpler block, it will provide a user to construct the code module representing the function of the depicted block. It appears covers the limitations recited within:

***A method for configuring a microcontroller, comprising:***

***displaying a first graphical user interface on a display device of a computer system, said first graphical user interface comprising a collection of virtual blocks in a design system*** (see figure 4);

***receiving at said computer system a selection of a user module, wherein said user module comprises information for implementing a function using a programmable physical block*** (Figure 4, selecting a 14 Bit ADC, part of Library modules or user modules (p. 3, include line 5));

***displaying on said display device a second graphical user interface operable for receiving user-specifiable information about said user module*** (Figure 4, selecting a 14 Bit ADC as a virtual block, and its module (p. 3) provided with user input parameters in the left area of the PSoC Designer);

***assigning a virtual block taken from said collection to said user module, wherein said virtual block corresponds to said programmable physical block*** (Figure 4, selecting a 14 Bit ADC, part of Library modules or user modules (p.3), and take the input parameters assigned by user into the module); ***and***

***constructing ~~completely~~ computer-generated source code that is loaded into a register of said programmable physical block to cause said programmable***

Art Unit: 2191

**physical block to implement said function, wherein constructing the computer-generated source code comprises substituting said user-specifiable information for generic information in a template file.**

Bindra shows each user module is designed by user code to present the function of that module (Figure 4). The function is represented by source code (user module) to cause its programmable physical block to implement the function. It shows the user uses the icon drag/drop higher level functions into approximate boxes. The configuration information is transferred into a register space (p.1). Bindra does not specific mention the source code is loaded into a register of said programmable physical block.

However, it should be noted that every source code that implement a virtual block is an executable code such as assembly language. There is no thing new for this.

Hamblen shows a programmable on chip design process (p. 36, Figure 11) using a CAD tool that takes design virtual blocks and constructs source code. The source code in form assemble or machine language that is automatically generated by a C compiler to mapped on to the Virtual blocks designed from the CAD tool (See Figure 1: VHDL Design entity (i.e. Virtual blocks) connected with automatically generated code generated by a C Compiler, mapped to virtual blocks at gate levels in the CAD tool) for obviously disclosing: “*automatically constructing source code*”

Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to know that, for an enablement, it requires every module in the PSoC Designer discussed by Bindra is automatically generated into executable “code”, and

Art Unit: 2191

**automatically connected** to a corresponding virtual block when a circuit is built in the system by a PSoc Designer, shown by Bindra. It is obvious because, in order to present a functionality of a circuit, a user can manually program a circuit in an assembly program with assigned parameters presenting element configuration (i.e. a source code that represents for a circuit); but it will **take days** for him to do so. For conforming to the availability of source code generation, tools and compiler is available for use at that time, thus it will generate automatically into assemble code and/or machine code for manual acts, as shown by Hamblen in Figure 11.

As per Claim 2: Bindra further discloses,

*The method of Claim 1, wherein said function comprises a pulse width modulator* (Bindra: See Figure 4, "User Module" that represents various Digital functions, and see P.2 line 36, "PWMs").

As per Claim 3: Bindra further discloses, *The method of Claim 1, wherein said function comprises a timer.* (Bindra: See Figure 4, refer to "User Module" that represents various Digital functions, and see P.2 line 36, "timers").

As per Claim 4: Bindra further discloses, *The method of Claim 1, wherein said function comprises an analog-to-digital converter* (Bindra: See Figure 4, refer to "User Module" that represents various Digital functions, and see P.2 line 35, "ADCs").

As per Claim 5: Bindra further discloses, *The method of Claim 1, wherein said function comprises a digital-to-analog converter* (Bindra: See Figure 4, refer to "User Module" that represents various Digital functions, and see P.2 line 35 "DACs").

Art Unit: 2191

As per Claim 6: Bindra further discloses, *The method of Claim 1, wherein said function comprises a counter* (Bindra: See Figure 4, refer to “User Module” that represents various Digital functions, and see P.2 line 36 “counters”).

As per Claim 7: Bindra further discloses, *The method of Claim 1, wherein said function comprises a signal amplifier*. (See Figure 4, refer to “User Module” that represents various Digital functions, and see P.2 line 33 “differential amplifiers”).

As per Claim 8: Bindra further discloses, *The method of Claim 1, wherein said function provides serial communication*. (See Figure 4, refer to “User Module” that represents various Digital functions, and see P.3, line 9, “serial transmitters/receivers”).

As per Claim 9: Bindra further discloses, *The method of Claim 1, wherein said collection is displayed as a two dimensional array of programmable analog virtual blocks and programmable digital virtual blocks*. (See collections in the right bottom section, which is *two-dimensional array*).

As per Claim 10: Bindra further discloses, *The method of Claim 1, wherein said assigning further comprises assigning a second virtual block to said user module* (See collections in the right bottom section, which is *two dimensional array*).

As per Claim 11: Bindra further discloses, *The method of Claim 1, wherein said source code comprises a symbolic name for a register address in said programmable physical block*. (Bindra: See page 2, lines 12-17 (‘register space that holds the configuration information’).

As per Claim 12: Bindra further discloses, *The method of Claim 11 wherein said symbolic name is derived from said function*. (See Bindra ‘User module’ in Figure 4, where user module

Art Unit: 2191

represents a circuit element. Each circuit element is a symbolic name function: e.g.: ADC, DAC, Timer, Counter, etc).

As per Claim 37: Bindra further discloses, *The method of Claim 1 wherein said user module is represented by first markup language data that includes information defining how configuration register for said microcontroller are be programmed in order to implement said function, and wherein said programmable physical is represented by second markup data that includes information defining physical addresses of said configuration registers.*

Official notice is taken that Markup language such as SGML, HTML, XML, is the language that is provided computer rendering. There is no invention on using the markup language for rendering, but compliant to the rules set forth by the language. This is well known, The Figure 4 of Bindra can be implemented by XML and its data, and the use of Markup languages is only to take the advantage of the availability that is designed for graphical rendering (The well-known HTML and XML is available in the Internet, Example, Applicants can refer to the website XML.com).

Therefore, it is obvious for an ordinary in the art when something like mark up language become standard for providing the graphical rendering; one will take the advantage to use it by including it for conforming to the availability.

To support this office notice: see Karayiannis "Using XML for Representation and Visualization of Elaborated VHDL-AMS Models", 2000, IEEE, pages 83-87.

As per Claim 36: Incorporated to the rejection of claim 1, Bindra and Hamblen further discloses

Art Unit: 2191

*The method of Claim 1 wherein said constructing the computer-generated source code further comprises:*

*reading the template file;*

*producing assembly, include, and header files from the template file,*

*wherein said user-specifiable information comprises information specific to said user module, information specific to said function*

*and information specific to a control parameter of said function;*

*compiling said assembly, include and header files to produce an executable file; downloading said executable file as a code block to a memory of said microcontroller; and executing said code block to configure said programmable block.*

as in the Figures 7, 9-11 (Hamblen). It is obvious, because the claimed recitation conforms to or complies with basis process of C compiler when it generates assembly language; i.e. a C program has “include statements”, or “header files”; therefore, every template created from the C compiler for an assembly program will include with “header files” such #include statement on its top. It is obvious because it is conforming to a C program → assembly program.

Bindra and Hamblen further in combined fail to disclose a very well-known markup language, that is usually available for graphical rendering; where the Markup language is usable for rendering virtual blocks such as the blocks shown in Figure 4.

Official notice is taken: the obviousness is as addressed as set forth in the rejection in the claim 37.

To support this office notice:

Art Unit: 2191

see Texas Instruments “TMS320C6000 Optimizing Compiler User’s Guide”, 4-2001, Texas Instruments, chapters 1: pages 1-7 and chapter 2: pages 1-44.

see Miguel “Implementation of an universal Boot Monitor for an ARM-based System”, 5-2000, TU Berlin, Germany, chapter 4: pages 53-100.

As per Claim 13: See the rationale addressed in Claim 1.

As per Claim 14: Regarding,

*“The method of Claim 13, wherein said automatically constructing further comprises:*

*computing a register address for a register within said programmable physical block;*

*determining a symbolic name for said register address, said symbolic name corresponding to*

*said user module and said circuit design; and substituting said symbolic name for a generic*

*name in said template assembly code”*. See page 2, lines 12-17 (‘register space that holds the

configuration information’) and page 6, lines 7- 13, (‘user modules are selected, pins are

assigned, and register mapping are establish’);

- *computing a register address for a register within said programmable block*: page 6, lines 7-13, referring “register mapping”

- *determining a symbolic name for said register address, said symbolic name corresponding to said user module and said circuit design*: page 2, lines 12-17, referring “holds the configuration information”.

- *substituting said symbolic name for a generic name in said template assembly code*: referring the code construction performed by the PSoC Designer.

As per Claim 16: regarding limitations of Claim 16.

Art Unit: 2191

See page 2, lines 12-17 ('register space that holds the configuration information') and page 6, lines 7- 13, ('user modules are selected, pins are assigned, and register mapping are establish') for

- *determining a symbolic name corresponding to said user module and said circuit design;*

referring "holds the configuration information".

- *computing a register address for a register within said programmable block;* referring "register mapping"

- *assigning said symbolic name to said register address; and placing said symbolic name into said assembly code in place of a generic name provided in said template assembly code file:*

referring the code construction performed by the PSoC Designer.

### ***Allowable Subject Matter***

5. Claims 26-27 are allowed.

6. Claim 38 is objected to as being dependent upon a rejected base claim, but would be allowable if rewritten in independent form including all of the limitations of the base claim and any intervening claims.

***Conclusion***

7. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ted T. Vo whose telephone number is (571) 272-3706. The examiner can normally be reached on 8:00AM to 4:30PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Wei Y. Zhen can be reached on (571) 272-3708.

The facsimile number for the organization where this application or proceeding is assigned is the Central Facsimile number 571-273-8300.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100. Information regarding the status of

Art Unit: 2191

an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

TTV  
November 20, 2009

/Ted T. Vo/  
Primary Examiner, Art Unit 2191